Variable Scope

#### **Learning Outcome Addressed**

   2. Identify whether a variable is in global, local, or block scope

### Scope Introduction

In JavaScript scope refers to the visibility of variables. Some variables are visible from anywhere in the program, and some are only visible within certain parts of the program.

While writing code, it is very useful to limit a variable's scope. If you set a variable's scope within a function, meaning that variable is not visible from outside the function, then you can make sure that changes within that function will not affect the rest of your program.

For example, look at the following code snippet and see if you can spot the issue.

function logMessage() {  
    // message can only be used in logMessage  
    var message = "declared inside function";  
    console.log("Inside function");  
    console.log(message);  
}  
// This console log will cause an error  
console.log(message);

The scope of the message variable is limited to inside the function, so attempting to reference the variable outside the function results in an error.

If we refactor the code, we can have a valid scope.

Valid scope:

//message is declared outside the function  
var message = "thinking global";  
   
logMessage();  
   
function logMessage() {  
    console.log("Inside function");  
    console.log(message);  
}  
   
console.log("Outside function");  
console.log(message);

### Task instructions

You task in this activity is to declare a variable inside the function yourPet defined in the scope.js file located inside the scope folder.

Note that there are two functions, myPet and yourPet, both return an animal. However, the value of the animal returned by yourPet should be different than the one myPet returns.

To accomplish this task, note the following:

* You cannot hard-code return 'cat' inside the yourPetfunction.
* yourPet must return a variable named animal.
* yourPet should not reassign the existing animal variable declared on the first line (in the global scope).

**Hint:** Remember that variables declared inside a function are within the function's scope.

Task

Given the myPet and yourPet functions, try to fix the scope that will make return different pet results.

<!DOCTYPE html>

<html>

<head>

 <title>Scope</title>

</head>

<body>

<p>To see your changes, refresh this window.</p>

 <script src="./scope.js"></script>

   <script src="./scope2.js"></script>

</body>

</html>

Scope.js

var animal = 'dog';

function myPet() {

  // You should not need to modify this function

  return animal;

}

function yourPet() {

  // The expectation for this function is to return `animal` just like the previous function

  // However, you cannot simply modify the existing variable declared in the global scope

  // Note: writing return 'cat' below will not work

  /////////////////////

  //your code goes here

  ////////////////////

  return animal;

}

//open your browser console to check the results

console.log('myPet result: ' + myPet());

console.log('yourPet result: ' + yourPet());

//Don't change this line

module.exports = { myPet: myPet, yourPet: yourPet };